UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/688,640	10/17/2003	Antti Kokkinen	200701903-2	1958

22879 7590 08/06/2010

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
3404 E. Harmony Road
Mail Stop 35
FORT COLLINS, CO 80528

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
08/06/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

JERRY.SHORMA@HP.COM
ipa.mail@hp.com
laura.m.clark@hp.com

Office Action Summary	Application No.	Applicant(s)	
	10/688,640	KOKKINEN, ANTTI	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *24 May 2010*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Applicant's amendment dated May 24, 2010, responding to the Non-Final Office action mailed February 24, 2010 provided in the rejection of claims 1-24; wherein claims 1-3, 12-13, 15, and 22-24 have been amended

Claims 1-24 remain pending in the application and which have been fully considered by the examiner.

Applicant's arguments with respect to claims currently amended have been fully considered but are moot in view of the new grounds of rejection – see *Abdallah et al.* and *Chiang* - arts made of record, as applied hereto.

2. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a).

Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Claim Rejections – 35 USC § 103(a)

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-24 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abdallah et al. (Pat. No. US 6,999,976 B2) (hereinafter 'Abdallah' - art made of record) in view of Ying-Hsin Robert Chiang et al. (US 2004/0062130 A1) (hereinafter 'Chiang' - art made of record)

5. **As to claim 1** (Currently Amended), Abdallah discloses a method for updating software in an electronic device (e.g., Col. 1, Lines 8-12 - … performing file system update … using a Java archive to encode file system update information; Col. 4, Lines 50-54 - … a persona digital assistant device … configured with ROM and/or flash ROM …), the method comprising:

- generating an update package (e.g., Col. 6, Lines 38-46 – the Delta class creates a delta JAR file … and places the difference information into a Delta JAR file (*interpreted as an update package*) … the resulting Delta JAR file is transported … "Application Update" …- emphasis added) for updating at least one software application being generated based upon difference information between the at least one software application and at

least one reference software installed on the electronic device (e.g., Col. 6, Lines 9-17 - … the computed difference information provides a means of updating a third file system tree ... so to update that installation to include fixes, updates ...; Col. 5, Lines 30-41 - … used to compare two file system trees and encode the resulting difference information into a specialized Java archive … - emphasis added);

- updating the at least one software application using the update package and the at least one reference software (e.g., Col. 1, Lines 8-12 - … performing file system update … using a Java archive to encode file system update information); and
- wherein the updating leaves the at least one reference software unchanged (e.g., since Abdallah compares two file system trees (e.g., see Fig. 4 and Col. 5, Lines 41-54), if there is no needs to update reference software, the resulting difference information will contain empty information; and, therefore it will leave a reference software unchanged)

Further, Abdallah discloses performing file system updates and using a Java archive to encode file system update information (e.g., Col. 1, Lines 7-12) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Updating Electronic Files Using Byte-Level File Differencing and Updating Algorithms*, Chiang discloses:

- the at least one reference software includes files common to a plurality of versions of the software application (e.g., [0051] - …the operating system

files, protocol stacks … communication libraries, display or LCD driver files …- emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Chiang into the Abdallah's system to further provide other limitations stated above in the Abdallah system.

The motivation is that it would further enhance the Abdallah's system by taking, advancing and/or incorporating the Chiang's system which offers significant advantages of an efficient representation of the byte-level differences in a delta file including meta-data along with actual data of replacement and/or insertion operations in a delta file; and the file differencing algorithm using a minimum number of bytes and pre-defined format or protocol, thereby providing a delta file optimized in space as once suggested by Chiang (e.g., [0030] – emphasis added)

6. **As to claim 2** (Currently Amended) (incorporating the rejection in claim 1), Chiang discloses the method, wherein the files common to a plurality of versions of the software application includes at least one of binaries, firmware code, dynamic link libraries (DLLs), (e.g., [0028] - … software files including dynamic link library files, shared object files ... firmware files, executable files …) and Abdallah further discloses JAVA archives (JAR files) (e.g., Col. 1, Line 22 – a Java archive (JAR))

7. **As to claim 3** (Currently Amended) (incorporating the rejection in claim 1), Abdallah discloses the method wherein the update package is performed based on the reference software installed on the electronic device instead of the software package such that only one update package is disseminated to update multiple different versions of the software application (e.g., Col. 4, Lines 61-65 - … a JAR file contains all the resources required to install and run a Java program in a single file … - emphasis added)

8. **As to claim 4** (Original) (incorporating the rejection in claim 1), Abdallah discloses the method further comprising updating multiple update versions of the at least one software application installed on the electronic device is performed using a single update package (e.g., Col. 4, Lines 61-65 - … a JAR file contains all the resources required to install and run a Java program in a single file … - emphasis added)

9. **As to claim 5** (Original) (incorporating the rejection in claim 1), Abdallah discloses the method further comprising installing the at least one software application and the at least one reference software as part of a single installation (e.g., Col. 4, Lines 61-65 - … a JAR file contains all the resources required to install and run a Java program in a single file … - emphasis added)

10. **As to claim 6** (Original) (incorporating the rejection in claim 1), Abdallah discloses the method further comprising updating the at least one reference

software and updating the at least one software application as part of a single update (e.g., Col. 4, Lines 61-65 - … a JAR file contains all the resources required to install and run a Java program in a single file … - emphasis added)

11. **As to claim 7** (Original) (incorporating the rejection in claim 1), Chiang discloses the method wherein the at least one software application comprises a plurality of software applications (e.g., [0033] - … all device software ranging from firmware to embedded applications ... - emphasis added), and the at least one reference software comprises a plurality of reference software (e.g., [0051] - …the operating system files, protocol stacks … communication libraries, display or LCD driver files …- emphasis added)

12. **As to claim 8** (Original) (incorporating the rejection in claim 7), Abdallah discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., Col. 5, Lines 61-67 – the two file system trees are identified as an "initial" file system tree and a "final" file system tree … the Delta class encodes information which is sufficient for updating the "initial" file system tree to be equivalent to the "final" file system tree); and
- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is not present, then installing the software

application and an associated reference software in a single update on the electronic device (e.g., Col. 8, Lines 16-27 - … a list is mode of the paths which are present in the list of the final file system tree and which are not present in the list of the initial file system tree … these are the paths files and directories which have been added … - emphasis added)

13. **As to claim 9** (Original) (incorporating the rejection in claim 7), Abdallah discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., Col. 5, Lines 61-67 – the two file system trees are identified as an "initial" file system tree and a "final" file system tree … the Delta class encodes information which is sufficient for updating the "initial" file system tree to be equivalent to the "final" file system tree);
- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is present then retrieving an update package for the software application needing updating (e.g., Col. 8, Lines 28-31 – a list is made of the paths which are present in both lists … these are the paths of files and directories which have neither been added nor removed, both which may have changed … - emphasis added); and

- installing the update package on the electronic device (e.g., Col. 6, Lines 13-17 - … to update that installation to include fixes, updates, and other modifications ... "application update" …)

14. **As to claim 10** (Original) (incorporating the rejection in claim 7), Abdallah discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., Col. 5, Lines 61-67 – the two file system trees are identified as an "initial" file system tree and a "final" file system tree … the Delta class encodes information which is sufficient for updating the "initial" file system tree to be equivalent to the "final" file system tree);
- determining if the update is needed immediately; and
- storing the update until the update is needed immediately (e.g., Col. 6, Lines 13-17 - … to update that installation to include fixes, updates, and other modifications ... "application update" …)

15. **As to claim 11** (Original) (incorporating the rejection in claim 10), Chiang discloses the method wherein when the update is determined to be needed immediately, then

- invoking an update agent to employ at least the stored update package and reference software (e.g., [0044] - … the upgrade client 206 ...

provides for the maintenance and upgrading of embedded device software components ... – emphasis added); and

- updating the software application with the update package (e.g., [0044] - … upgrades a software component …)

16. **As to claim 12** (Currently Amended), Abdallah discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., Col. 1, Lines 8-12 - … performing file system update … using a Java archive to encode file system update information; Col. 4, Lines 50-54 - … a persona digital assistant device … configured with ROM and/or flash ROM …);
- a software delivery device for receiving and installing a reference software to the electronic device if the electronic device does not have the reference software previously installed (e.g., Col. 8, Lines 16-27 - … a list is mode of the paths which are present in the list of the final file system tree and which are not present in the list of the initial file system tree … these are the paths files and directories which have been added … - emphasis added); and
- the software delivery device receiving and delivering at least one update package to the electronic device (e.g., Col. 6, Lines 38-46 – the Delta class creates a delta JAR file … and places the difference information into a Delta JAR file (*interpreted as an update package*) … the resulting Delta

JAR file is transported … "Application Update" … - emphasis added), wherein the at least one update package is based on differences between at least one application software and the reference software, and the reference software facilitates (e.g., Col. 6, Lines 9-17 - … the computed difference information provides a means of updating a third file system tree ... so to update that installation to include fixes, updates ...; Col. 5, Lines 30-41 - … used to compare two file system trees and encode the resulting difference information into a specialized Java archive … - emphasis added), using the at least one update package, at least one update to the application software installed on the electronic device (e.g., Col. 1, Lines 8-12 - … performing file system update … using a Java archive to encode file system update information), and wherein the updating leaves the reference software unchanged (e.g., since Abdallah compares two file system trees (e.g., see Fig. 4 and Col. 5, Lines 41-54), if there is no needs to update reference software, the resulting difference information will contain empty information; and, therefore it will leave a reference software unchanged)

Further, Abdallah discloses performing file system updates and using a Java archive to encode file system update information (e.g., Col. 1, Lines 7-12) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Updating Electronic Files Using Byte-Level File Differencing and Updating Algorithms*, Chiang discloses:

- the reference software includes a plurality of shared files with the application software (e.g., [0051] - …the operating system files, protocol stacks … communication libraries, display or LCD driver files …- emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Chiang into the Abdallah's system to further provide other limitations stated above in the Abdallah system.

The motivation is that it would further enhance the Abdallah's system by taking, advancing and/or incorporating the Chiang's system which offers significant advantages of an efficient representation of the byte-level differences in a delta file including meta-data along with actual data of replacement and/or insertion operations in a delta file; and the file differencing algorithm using a minimum number of bytes and pre-defined format or protocol, thereby providing a delta file optimized in space as once suggested by Chiang (e.g., [0030] – emphasis added)

17. **As to claim 13** (Currently Amended) (incorporating the rejection in claim 12), Chiang discloses the system wherein the plurality of shared files include binaries and firmware code (e.g., [0028] - … software files including dynamic link library files, shared object files ... firmware files, executable files …)

18. **As to claim 14** (Original) (incorporating the rejection in claim 12), Chiang discloses the system further comprising an update generating system, the update generating system comprising a loader manager, the loader manager:

- managing loading of application software and application software version updates from the software delivery device (e.g., [0043] - … the manager of client device configuration data, indentifies the current versions of embedded software components of the requesting device 104, identifies and transfers appropriate delta files …);
- employing a loader from a loader module (e.g., [0062] – … the upgrade client then downloads the delta file … the target EBSC to be updated …); and
- employing security services to authenticate software being delivered (e.g., [0041] - … capabilities including authenticating … - emphasis added)

19. **As to claim 15** (Currently Amended) (incorporating the rejection in claim 12), Chiang discloses the system wherein the plurality of shared files include dynamic link libraries (DLLs) (e.g., [0028] - … software files including dynamic link library files, shared object files ... firmware files, executable files …) and Abdallah further discloses JAVA archives (JAR files) (e.g., Col. 1, Line 22 – a Java archive (JAR)); loader manager further comprises an installation agent for installing application software and downloading files from the software delivery device (e.g., [0044] - … the upgrade client 206 ... provides for the maintenance and upgrading of embedded device software components ... – emphasis added)

20. **As to claim 16** (Original) (incorporating the rejection in claim 16), Chiang discloses the system wherein the loader manager is adapted to:

- identify an application software needing updating (e.g., Col. 5, Lines 61-67 – the two file system trees are identified as an "initial" file system tree and a "final" file system tree … the Delta class encodes information which is sufficient for updating the "initial" file system tree to be equivalent to the "final" file system tree);
- identify whether reference software associated with the application software needing updating exists (e.g., Col. 8, Lines 16-27 - … a list is mode of the paths which are present in the list of the final file system tree and which are not present in the list of the initial file system tree … these are the paths files and directories which have been added … - emphasis added); and
- coordinating an update of the application software and an associated reference software in a single update (e.g., Col. 4, Lines 61-65 - … a JAR file contains all the resources required to install and run a Java program in a single file … - emphasis added)

21. **As to claim 17** (Original) (incorporating the rejection in claim 14), Chiang discloses the system wherein the loader manager is adapted to:

- retrieve the update package (e.g., [0088] - … the upgrade client then downloads the delta file block from the update server into device RAM …);

- access contents of the update package; and
- verify the update package (e.g., [0097] - … use Cyclic Redundancy Check (CRC) to detect any corruption of a transmitted delta file)

22. **As to claim 18** (Original) (incorporating the rejection in claim 14), Chiang discloses the system wherein the loader manager is adapted to determine immediacy of a needed update for a particular application software (e.g., [0047] - … critical software … non-critical EBSCs …)

23. **As to claim 19** (Original) (incorporating the rejection in claim 12), Chiang discloses the system wherein the software delivery device is one of a server, a CDROM, and a network (e.g., Fig. 2, Upgrade Server 204; communication infrastructure 212)

24. **As to claim 20** (Original) (incorporating the rejection in claim 12), Chiang discloses the system wherein the electronic device is one of a computer, a digital phone, and a digital camera (e.g., Fig. 2, client device 104)

25. **As to claim 21** (Original), Abdallah discloses a method for updating software in an electronic device the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between first and second software versions (e.g.,

Col. 6, Lines 9-17 - … the computed difference information provides a means of updating a third file system tree ... so to update that installation to include fixes, updates ...; Col. 5, Lines 30-41 - … used to compare two file system trees and encode the resulting difference information into a specialized Java archive … - emphasis added); and

- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information between first and third software versions (e.g., see the rejection contained in first bullet above);

Further, Abdallah discloses an advantageous mechanism for encoding file system update information as a special Java archive file to hold difference information (see Col. 7, Lines 13-31 for the details) suitable for use to update a file system tree; the specialized Java archives hold not only the file system difference information, but also a Java extractor class and manifest information which specifies that the extractor class is the main class to be executed (e.g., Col. 5, Lines 16-29 – emphasis added); the Delta class (see Col. 7, Line 63 through Col. 15, Line 16 for details) encodes information which is sufficient for system tree update (e.g., Col. 5, Lines 61-67)

Therefore, it would be apparent to those of ordinary skill in the art to modify the Abdallah's teaching to:

- generate a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages;

and update the at least one software application using the third update package

Furthermore, Abdallah discloses performing file system updates and using a Java archive to encode file system update information (e.g., Col. 1, Lines 7-12) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Updating Electronic Files Using Byte-Level File Differencing and Updating Algorithms*, Chiang discloses:

- the at least one reference software includes files common to a plurality of versions of the software application (e.g., [0051] - …the operating system files, protocol stacks … communication libraries, display or LCD driver files …- emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Chiang into the Abdallah's system to further provide other limitations stated above in the Abdallah system.

The motivation is that it would further enhance the Abdallah's system by taking, advancing and/or incorporating the Chiang's system which offers significant advantages of an efficient representation of the byte-level differences in a delta file including meta-data along with actual data of replacement and/or insertion operations in a delta file; and the file differencing algorithm using a minimum number of bytes and pre-defined format or protocol, thereby providing a delta file optimized in space as once suggested by Chiang (e.g., [0030] – emphasis added)

26. **As to claim 22** (Currently Amended), Abdallah discloses a method for updating software in an electronic device, the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between a first software version and a reference software corresponding to the at least one software application (e.g., Col. 6, Lines 9-17 - … the computed difference information provides a means of updating a third file system tree ... so to update that installation to include fixes, updates ...; Col. 5, Lines 30-41 - … used to compare two file system trees and encode the resulting difference information into a specialized Java archive … - emphasis added);
- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information a second software version and the reference software corresponding to the at least one software application (e.g., Col. 6, Lines 9-17 - … the computed difference information provides a means of updating a third file system tree ... so to update that installation to include fixes, updates ...; Col. 5, Lines 30-41 - … used to compare two file system trees and encode the resulting difference information into a specialized Java archive … - emphasis added);

- updating the at least one software application using the third update package (e.g., Col. 1, Lines 8-12 - … performing file system update … using a Java archive to encode file system update information)

Further, Abdallah discloses an advantageous mechanism for encoding file system update information as a special Java archive file to hold difference information (see Col. 7, Lines 13-31 for the details) suitable for use to update a file system tree; the specialized Java archives hold not only the file system difference information, but also a Java extractor class and manifest information which specifies that the extractor class is the main class to be executed (e.g., Col. 5, Lines 16-29 – emphasis added); the Delta class (see Col. 7, Line 63 through Col. 15, Line 16 for details) encodes information which is sufficient for system tree update (e.g., Col. 5, Lines 61-67)

Therefore, it would be apparent to those of ordinary skill in the art to modify the Abdallah's teaching to:

- generate a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages; and update the at least one software application using the third update package

Further, Abdallah discloses performing file system updates and using a Java archive to encode file system update information (e.g., Col. 1, Lines 7-12) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Updating Electronic Files Using Byte-Level File Differencing and Updating Algorithms*, Chiang discloses:

- wherein the reference software includes files common to the at least one software application and to the second software version (e.g., [0051] - …the operating system files, protocol stacks … communication libraries, display or LCD driver files …- emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Chiang into the Abdallah's system to further provide other limitations stated above in the Abdallah system.

The motivation is that it would further enhance the Abdallah's system by taking, advancing and/or incorporating the Chiang's system which offers significant advantages of an efficient representation of the byte-level differences in a delta file including meta-data along with actual data of replacement and/or insertion operations in a delta file; and the file differencing algorithm using a minimum number of bytes and pre-defined format or protocol, thereby providing a delta file optimized in space as once suggested by Chiang (e.g., [0030] – emphasis added)

27. **As to claim 23** (Currently Amended) (incorporating the rejection in claim 22), Chiang discloses the method, wherein the files include at least one of binaries, firmware code, dynamic link libraries (DLLs) (e.g., [0028] - … software files including dynamic link library files, shared object files ... firmware files,

executable files …) and Abdallah further discloses JAVA archives (JAR files) files) (e.g., Col. 1, Line 22 – a Java archive (JAR))

28. **As to claim 24** (Currently Amended), Abdallah discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., Col. 1, Lines 8-12 - … performing file system update … using a Java archive to encode file system update information; Col. 4, Lines 50-54 - … a persona digital assistant device … configured with ROM and/or flash ROM …);
- a first update package generator for generating update packages (e.g., Col. 6, Lines 38-46 – the Delta class creates a delta JAR file … and places the difference information into a Delta JAR file (*interpreted as update packages*) … the resulting Delta JAR file is transported … "Application Update" … - emphasis added) based upon difference information between a version of software and a reference software corresponding to at least one software application (e.g., Col. 6, Lines 9-17 - … the computed difference information provides a means of updating a third file system tree ... so to update that installation to include fixes, updates ...; Col. 5, Lines 30-41 - … used to compare two file system trees and encode the resulting difference information into a specialized Java archive … - emphasis added);

Further, Abdallah discloses an advantageous mechanism for encoding file system update information as a special Java archive file to hold difference information (see Col. 7, Lines 13-31 for the details) suitable for use to update a file system tree; the specialized Java archives hold not only the file system difference information, but also a Java extractor class and manifest information which specifies that the extractor class is the main class to be executed (e.g., Col. 5, Lines 16-29 – emphasis added); the Delta class (see Col. 7, Line 63 through Col. 15, Line 16 for details) encodes information which is sufficient for system tree update (e.g., Col. 5, Lines 61-67)

Therefore, it would be apparent to those of ordinary skill in the art to modify the Abdallah's teaching to:

- a second update package generator for generating update packages based upon difference information between different update packages; and
- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device.

Furthermore, Abdallah discloses performing file system updates and using a Java archive to encode file system update information (e.g., Col. 1, Lines 7-12) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Updating Electronic Files Using Byte-Level File Differencing and Updating Algorithms*, Chiang discloses:

- wherein the reference software includes at least one of a plurality of shared binaries, firmware code, dynamic link libraries (DLLs) (e.g., [0028] - … software files including dynamic link library files, shared object files ... firmware files, executable files …) and Abdallah furthermore discloses JAVA archives (JAR files) with the at least one software application (e.g., Col. 1, Line 22 – a Java archive (JAR))

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Chiang into the Abdallah's system to further provide other limitations stated above in the Abdallah system.

The motivation is that it would further enhance the Abdallah's system by taking, advancing and/or incorporating the Chiang's system which offers significant advantages of an efficient representation of the byte-level differences in a delta file including meta-data along with actual data of replacement and/or insertion operations in a delta file; and the file differencing algorithm using a minimum number of bytes and pre-defined format or protocol, thereby providing a delta file optimized in space as once suggested by Chiang (e.g., [0030] – emphasis added)

Conclusion

29. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is

571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Examiner, Art Unit 2192

/Michael J. Yigdall/
Primary Examiner, Art Unit 2192